Automatic Transformation of Raw Clinical Data Into Clean Data Using Decision Tree Learning Combining with String Similarity Algorithm*

Jian Zhang

School of Computing, University of Dundee
Perth Road, Dundee, UK
j.s.zhang@dundee.ac.uk

Abstract

It is challenging to conduct statistical analyses of complex scientific datasets. It is a time-consuming process to find the relationships within data for whether a scientist or a statistician. The process involves preprocessing the raw data, the selection of appropriate statistics, performing analysis and providing correct interpretations, among which, the data pre-processing is tedious and a particular time drain. In a large amount of data provided for analysis, there is not a standard for recording the information, and some errors either of spelling, typing or transmission. Thus, there will be many expressions for the same meaning in the data, but it will be impossible for analysis system to automatically deal with these inaccuracies. What is needed is an automatic method for transforming the raw clinical data into data which it is possible to process automatically.In this paper we propose a method combining decision tree learning with the string similarity algorithm,which is fast and accuracy to clinical data cleaning. Experimental results show that it outperforms individual string similarity algorithms and traditional data cleaning process.

1998 ACM Subject Classification H.3.3 Information Search and Retrieval, I.2.6 Learning

Keywords and phrases Raw Clinical Data, Decision Tree Learning, String Similarity Algorithm

Digital Object Identifier 10.4230/OASIcs.ICCSW.2015.87

1 Introduction

Cancer research has become a greatly data rich environment. Several data analysis packages can be used for analyzing the data like SPSS (Statistical Product and Service Solutions) [1], Minitab [9]. However, before analyzing the data, it needs to be preprocessed to fix the errors, misspelling of the raw data and transform the raw data into an uniform data [15], making it fit-for-purpose. This process is both time-consuming and tedious. For example, a specific binary variable may require only the entries 'Yes' or 'No' in a large data column. However, 'Yes' may have been entered onto a spreadsheet as Yes, yes, Y, y, yES, 1, or been misspelled as Yed, yef, y es, y e s (note the inappropriate use of spacing), etc.. Clearly, there are an infinite number of possibilities of entering this 3-letter word incorrectly, and each of these entries is treated as separate entities by a computer program.

Obviously, we can see that there are only three possible answers in this data like the first column of the Table 1 – Yes, No or null. However, the errors include spelling, typing or transmission [2]. Besides, there is not a standard for scientists to collect data and record

* This work was partially supported by Henry Lester Trust.

2015 Imperial College Computing Student Workshop (ICCSW 2015).
Editors: Claudia Schulz and Daniel Liew; pp. 87–94
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Table 1 The example of raw data.

Clean data	Raw data
yes	yes, Yed, yef, Y...
no	No, N, not...
null	don't know, waiting for lab...

them. Thus, the collected data will probably be the raw data which includes some errors like that in the second column of the Table 1. It is difficult for the system to directly deal with these inaccuracies for statistical analysis.

Currently statisticians will manually change these entries into a uniform string or a number allowing the system to do further analysis. Potter's Wheel [16], Google Refine [17, 5] are interactive data cleaning system, which can be used to clean the data up and transform the data from one form into another. Therefore, it will save time for statisticians if an automated method can be used instead of a manual operation [4]. During the transformation, the statisticians usually amend these data using their previous experience.

Decision tree learning is a common algorithm of machine learning [10]. It will set some rules via previous experience to build a tree, and then predict the result using the built tree. Its process is similar to the operation of transformation by statisticians mentioned above. According to this feature and the necessity of data classification, decision tree learning has been chosen to clean the raw data automatically.

Meanwhile,the comparison between the new entry and the previous data is another important method to transform the raw clinical data to clean data during the manual operation [6]. Thus, it is worth attempting the string similarity algorithm for this research.

In the raw clinical data, there is ordinal data and continuous data. The ordinal data hold larger percentage in the research data so this paper will explore the research in the ordinal data, especially two-category data.

In this paper, decision tree learning algorithms and string similarity algorithm will be separately investigated to transform the raw clinical data to clean data and their features will be explored. After that, the result of the two methods will be analyzed. Finally, the two method will be combined as their features for the raw clinical data.

2 Decision Tree Learning

In general,training, validation and comparison with training data set will be the process of making a prediction in machine learning. Decision tree learning as one of the machine learning algorithm is a method for approximating discrete-valued functions [10], which is one of the most popular inductive algorithms. Decision tree learning is a method to classify different values under different attributes. It can set the rules based on the past data to classify the new raw data to the clean data, whose process is the same as the manual data pre-processing.

2.1 Decision Tree Representation

Decision trees classify data through arranging the case from the tree root to a leaf. Every internal node (not leaf node) on the tree stands for a test of the value of the attribute; the branch represents the result of the test; every leaf represents a classification categories. In short, decision trees are similar to the tree structure of the workflow, adopting the top-down

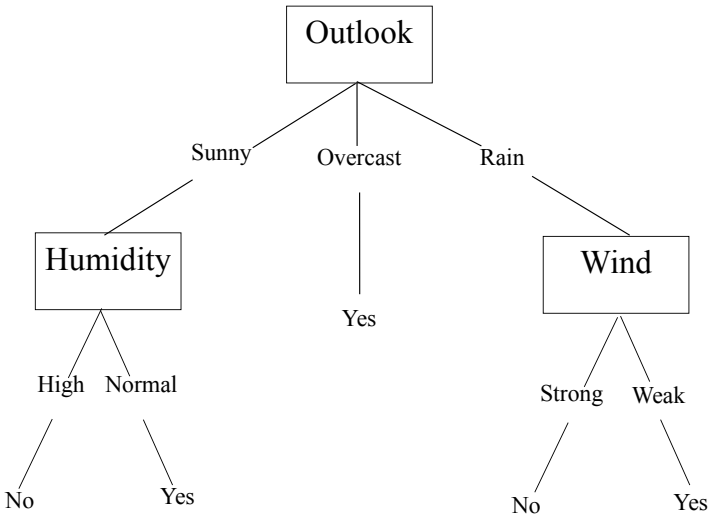

Figure 1 A simple decision tree example from [13].

inductive method. It begins from the root node, and then tests the value of the attribute in the internal nodes, next, confirms the matched branch based on the value of the attribute, finally, gets the result in the leaf node [10].

Figure 1 represents a typical learned decision tree. It describes the relationships between the weather and playing tennis [10]. In the figure, Yes equals play tennis, and No equals not to play tennis. There are three attributes, outlook, humidity and windy. Each attribute has their own value. For example, {outlook=rain; windy=strong; result =No} is a branch of the tree.

Generally, decision trees represent a disjunction of conjunctions of constraints on the attribute values of instances [10]. Each path from the root to a leaf corresponds to these disjunctions of conjunctions. In order to classify the raw clinical data, it can test the attribute values based on the structure of the decision tree, from the tree root down to the child node, and finally to the leaf node. And the corresponding category of the leaf node will be the category of the data object.

2.2 Decision Tree Learning Algorithm

ID3 was proposed by Quinlan in 1986 [13]. It is a representation of a decision tree algorithm and most of the decision tree algorithms are achieved based on improvements to it. It adopts the divide-and-conquer strategy and uses information gain as the standard to choose the attribute at the different levels of the decision tree in order that it can collect the most information of the categories about the test records in the process of testing each non-leaf node.

And its demerits are that it can only process the discrete attribute, and sometimes it is not the best to process the attribute with lots of values.

As a result of some problems with the ID3 algorithms in the practical application, Quinlan proposed the C4.5 algorithm [12], strictly speaking, which is just an improved algorithm of ID3.C4.5 algorithm inherits the advantages of the ID3 algorithm, and improves several aspects of the ID3 algorithm. C4.5:

- It uses information gain ratio to choose an attribute so that it overcomes the weakness

Table 2 Probability of transforming the test data to clean data using different decision tree algorithm.

Decision Tree	Unknown Entry (%)	Total (%)	Time (ms)
ID3	4	75	109
C4.5	27	81	109

that the system tends to choose the attribute with more values when using information gain.

- It adds prune in the process of constructing the tree.
- It can complete the discretization of the continuous attributes.
- It can process the incomplete data [14].

2.3 Experiments and Results

Weka (Waikato Environment for Knowledge Analysis) is a popular suite of machine learning software written in Java, which contains a collection of algorithms for data analysis and predictive modeling [18]. The algorithm ID3 and C4.5 are also coded in this software, and the detailed results including the possibility of the prediction will be calculated as well, so it will be adopted in testing the data.

The raw clinical data in this research is about breast cancer research from the website [3]. As mentioned in the introduction section, the two-category data will be undertaken in this experiment. The value of data is similar as the data in the Table 1. It will be divided into two parts, and one part is as training data set and the other part is as testing data set. In the testing data set, the data can be found in the training dataset, which is called the existing data; the other part is called the unknown data, which takes up 26%.

Table 2 shows that the probability of the correct transformation using the algorithm ID3 and C4.5 for the existing data, the unknown data and total. All the existing entries have been cleaned in the process. Compared with the probability of the correct transformation for the existing data, the percentage for the unknown data is quite low, which is 4% and 27% of all the unknown data respectively for ID3 and C4.5. Total results for correct transformation is 75% and 81% respectively.

All in all, decision tree learning have a good performance for the existing data transformation, but have a low performance on the unknown data [19]. The efficiency of the transformation mainly depends on the percentage of the existing data in the whole testing data set.

3 String Similarity Algorithm

3.1 String Similarity Algorithms

In this research, it is necessary to calculate the string similarity of the two strings when the entry is transformed from the raw data to processable data by the system. String-based similarity has a long history. Levenshtein proposed the edit distance, which is widely used for string similarity through calculating the minimum number of insertions, substitutions and deletions between two strings [8].The Levenshtein distance will be calculated once when add, delete or substitute have been done once during transformation. It provides a numeric

Table 3 Probability of transforming the test data to clean data using different string similarity algorithms.

Decision Tree	Unknown Entry (%)	Total (%)	Time (ms)
C4.5	27	81	109
Levenshtein Distance	58	89	1045
Needleman-Wunsch	73	89	846
Jaro-Winkler distance	73	91	907
N-W + Len	73	93	900

approach for transformation. For example, the Levenshtein distance between 'Yef' and 'Yes' is 1, because only one edit should be done, substitution of 's' for 'k'.

Needleman and Wunsch [11] extended the model to allow contiguous sequences of mismatched characters, or gaps, in the alignment of two strings, and described a general dynamic programming method for computing edit distance. For example, the score of Needleman and Wunsch between 'Yef' and 'Yes' is 2, since there are two matching letters 'Ye' between them.

Jaro-Winkler distance find the approximating string matching by means of calculating the number of matching characters and the number of transpositions. And a prefix scale is added in this method as well [7]. For example, the score between 'Yef' and 'Yes' is 0.82, and the score between 'Yef' and 'Tef' is 0.77. Even though the two group of string both have two matching letters, the first group have the prefix matching. Hence, 'Yef' is more similar as 'Yes' rather than 'Tef' based on this algorithm.

The Needleman-Wunsch algorithm is used to calculate the longest common substring (LCS). We can consider the effect of the length (Len) of the string in the N-W algorithm to improve it. For example,the score between 'Yef' and 'Yes' is 0.67, and the score between 'Yef' and 'Yest' is 0.57. Even though the two group of string both have two matching letters, the first group have the shorter string. Hence, 'Yef' is more similar as 'Yes' rather than 'Yest' based on this algorithm.

3.2 Experiments and Results

In this experiment, the training data set and the testing data set in the previous section will be used again. Because the algorithm C4.5 gets a higher performance and is suitable for the continuous data, which can be used in the future research, the result of C4.5 will be added to the result of this experiment for analysis.

Table 3 represents that the same testing elements as previous experiment. For all the existing entries, Levenshtein distance, Jaro-Winkler distance and improved N-W have the 100% as the same as the C4.5 gets. The Needleman-Wunsch algorithm gets 97%. The string similarity algorithms get a better results for the unknown data rather than the algorithm C4.5. The result for Levenshtein distance transforming the unknown is more than 2 times as the algorithm C4.5. The results for rest algorithm are much higher, which is 73% of the unknown data. Because of the larger improvements for the unknown data, the string similarity algorithm have a lot improvements for transforming the raw data to clean over the C4.5 algorithm.

Overall, the string similarity have a higher performance for the unknown data transformation. However, it is found that the time for running the string similarity algorithm is much slower than the algorithm C4.5 during the transformation.

Table 4 Probability of correct prediction for different string similarity algorithms combined with decision tree learning.

Decision Tree	Unknown Entry (%)	Total (%)	Time (ms)
C4.5	27	81	109
C4.5 + Lev	58	89	342
C4.5 + N-W	65	91	292
C4.5 + J-W	73	93	308
C4.5 + N-W + Len	73	93	305

4 Decision Tree Learning Combined with String Similarity Algorithm

According to the two previous experiments, the algorithms C4.5 have a low performance for the unknown data transformation but have fast process whilst the string similarity algorithm has a higher performance for the unknown data but is much slower. Thus, the combination of the two algorithms is worth exploring based on their features and performance.

The string similarity algorithm Levenshtein distance (Lev), Needleman-Wunsch (N-W), Jaro-Winkler distance (J-W) and a improved string similarity algorithm based on Needleman-Wunsch (NW-Len) will be undertaken to process the incorrect prediction in the following experiment combined with the algorithm C4.5.

This experiment will be undertaken the same training data set and testing data set as the previous experiment.

Table 4 shows that the probability of correct prediction for different combining algorithms and includes that for the algorithm C4.5 for comparison. The second row of the table shows the results of decision tree prediction. The other rows represent the results of each string similarity algorithms combining with the decision.The result for transforming the unknown data is the same as them without the combination. The improved NW-Len combined with the algorithm C4.5 share the highest percentage (73%) with the J-W distance combined with the algorithm C4.5. And they both get the highest percentage (93%) of the total data transformation. The result for the N-W algorithm and the Levenshtein distance combined with the algorithm C4.5 is 91% and 89% respectively, which is higher than the result of the algorithm C4.5 (81%).

To sum up, the results for transforming the raw clinical data to the processable data have improved significantly after combining the decision tree learning algorithm with string similarity algorithms, especially with the Jaro-Winkler distance and the modified Needleman-Wunsch algorithm to compare with decision tree learning.And the combining algorithm get much faster than the individual string similarity algorithm.

5 Conclusion and Future Work

This paper attempts to find an approach to reduce the manual operation for tedious and repeated data transformation work. To sufficiently introduce the performance of the decision tree learning algorithm combined with the string similarity algorithm, this paper firstly introduces the decision tree learning algorithm and tests its performance for transformation from the raw clinical data to clean data. The results represent that it has high performance for the existing entry but low performance for the unknown data. The testing process is fast. Secondly, this paper undertakes the string similarity algorithm to test in the same way. The

results demonstrate that it has a higher performance for the unknown data. However, the process is slower than the decision tree algorithm. Finally, this paper tests the efficiency of the combining algorithm. The results show that it has the high performance for the existing data as the algorithm C4.5 and the same performance for the unknown data as the string similarity algorithms. And the process is slower than the algorithm C4.5 but quite faster than the string similarity algorithm.

Even though the results demonstrate that the decision tree algorithm combined with the string similarity algorithm can to some extent, automatically transforming the raw clinical data into the clean data in order to reduce the manual operation, there is still plenty of challenge in the further research. Firstly, the time-consuming may be a problem for a very large data set transformation when the training data set is not rich. This paper investigates the transformation for two-category data. Secondly, it can be extended to the more than two category data. Thirdly, once the raw data set has been transformed by the combining algorithm, the correct transformed the unknown entry can be considered to update the training data set, and then a new decision tree will be built so that maybe the next transformation process will be faster. What's more, the process of transformation is not completely automated in this paper. There is still a lot space for improving.

References

1 Alan C Acock. Sas, stata, spss: A comparison. alan c. acock. *Journal of Marriage and Family*, 67(4):1093–1095, 2005.

2 Cyril N Alberga. String similarity and misspellings. *Communications of the ACM*, 10(5):302–313, 1967.

3 Clinical. `https://www.clinicalstudydatarequest.com/`. Accessed: 25 June 2015.

4 Mita K Dalal and Mukesh A Zaveri. Automatic text classification: a technical review. *International Journal of Computer Applications*, 28(2):37–40, 2011.

5 Google. `https://code.google.com/p/google-refine/`. Accessed: 25 June 2015.

6 D Gussfield. Algorithms on strings, trees, and sequences. *Computer Science and Computional Biology (Cambrigde, 1999)*, 1997.

7 Matthew A Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. *Journal of the American Statistical Association*, 84(406):414–420, 1989.

8 Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10(8):707–710, 1966.

9 Ruth Meyer and David Krueger. *Minitab guide to statistics*. Prentice-Hall, Inc., 1997.

10 Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

11 Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

12 J Quinlan. R.(1993) c4. 5: Programs for machine learning, 1993.

13 J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

14 J. Ross Quinlan. Improved use of continuous attributes in c4. 5. *Journal of artificial intelligence research*, pages 77–90, 1996.

15 Erhard Rahm and Hong Hai Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3–13, 2000.

16 Vijayshankar Raman and Joseph M Hellerstein. Potter's wheel: An interactive data cleaning system. In *VLDB*, volume 1, pages 381–390, 2001.

17 Seth Van Hooland, Ruben Verborgh, Max De Wilde, Johannes Hercher, Erik Mannens, and Rik Van de Walle. Free your metadata: Integrating cultural heritage collections through google refine reconciliation. *Pre-submission paper available on. http://freeyourmetadata.org/publications/freeyourmetadata. pdf*, 2011.

18 Ian H Witten, Eibe Frank, Leonard E Trigg, Mark A Hall, Geoffrey Holmes, and Sally Jo Cunningham. Weka: Practical machine learning tools and techniques with java implementations, 1999.

19 Jian Zhang, Karen Petrie, and Tingting Yu. Automatic transformation of raw clinical data into clean data using decision tree learning. *LMT*, 84(91):0–2344.